Impact of Evolutionary Community Detection Algorithms for Edge Selection Strategies

Paolo Barsocchi
Istituto di Scienza e Tecnologie dell'Informazione ISTI-CNR National Council of Research, Pisa, Italy
paolo.barsocchi@isti.cnr.it
https://orcid.org/0000-0002-6862-7593

Dimitri Belli
Department of Computer Science University of Pisa, Italy
dimitri.belli@di.unipi.it,
https://orcid.org/0000-0003-1491-6450

Stefano Chessa
Department of Computer Science and Istituto di Scienza e Tecnologie dell'Informazione ISTI-CNR University of Pisa, Italy
stefano.chessa@unipi.it,
https://orcid.org/0000-0002-1248-9478

Luca Foschini
Dipartimento di Informatica: Scienza e Ingegneria University of Bologna, Bologna, Italy
luca.foschini@unibo.it,
https://orcid.org/0000-0001-9062-3647

Michele Girolami
Istituto di Scienza e Tecnologie dell'Informazione ISTI-CNR National Council of Research, Pisa, Italy
michele.girolami@isti.cnr.it,
https://orcid.org/0000-0002-3683-7158

***Abstract*— The combination of the edge computing paradigm with Mobile CrowdSensing (MCS) is a promising approach. However, the selection of the proper edge nodes is a crucial aspect that greatly affects the performance of the extended architecture. This work studies the performance of an edge-based MCS architecture with ParticipAct, a real-word experimental dataset. We present a community-based edge selection strategy and we measure two key-metrics, namely latency and the number of requests satisfied. We show how they vary by adopting three evolutionary community detection algorithms, TILES, Infomap and iLCD configured by changing several configuration settings. We also study the two metrics, by varying the number of edge nodes selected so that to show its benefit.**

Keywords—CrowdSensing, Multi-access Edge Computing, Mobile Edge, Community Detection.

I. Introduction

The beginning of the new century has been characterized by a widespread of mobile and wearables devices without precedent. Such phenomenon has also fostered the development of massive sensing techniques like Mobile CrowdSensing (MCS) combined with distributed network architectural concepts, such as Multi-access Edge Computing (MEC). The MEC paradigm can be implemented with a set of intermediate nodes reducing the network traffic to and from the Cloud and providing, at the same time, location-based services to devices in proximity. Such paradigm can be adopted for an MCS scenario. More specifically, the back-end of a MCS architecture collects data from the crowd by propagating *sensing tasks* to end-devices . Data collected from devices are, in turn, uploaded to the Cloud. In this scenario, our goal is to reduce as much as possible those network interactions not required between the back-end and the sensing devices. To this purpose, we consider that the MEC paradigm can be here adopted to improve the performance of the MCS architecture.

With this paper, we study the performance of a MCS architecture extended with a set of mobile MECs acting as a data retrieval layer. A mobile MEC (M^2EC) is a proxy for other devices. The M^2EC plays a primary role in a MCS architecture, since it allows to increase the reliability of the data collection process, supporting the sensing activity of MCS mobiles and by providing a first level of data aggregation. In our previous studies, we demonstrated the effectiveness of using M^2ECs instead of standard MEC proxies [1, 2][6]. In particular, we exploited the social behavior of users joining an MCS measurement campaign in order to optimize the design of a MCS – MEC architecture. In this work, we still focus on such approach and we observe that people tend to periodically interact and to cluster with others according to their social attitude. This is typical of commuters that daily meet at the train station, or employees of the same company that meet in the same office. In these cases, people form a so-called community, namely a set of users co-located at the same time. When people form a community, their devices can interact directly by means of short-range network interfaces such as Wi-Fi Direct, LTE Direct network interfaces, but also Bluetooth communications [9]. We tested our MCS – MEC architecture by using the ParticipAct dataset [3], which reproduces the user's mobility in an urban area. Communities are detected by using 3 algorithms specifically designed for dynamic networks, namely Infomap [4], TILES [5], and iLCD [7]. The resulting communities are used to select the M^2ECs and to measure two evaluation metrics: latency and number of requests satisfied. All tests have been performed through an ad hoc MCS simulator able to reproduce the interaction from the back-end to the crowd's devices. We show how the community detection algorithms affect the metrics, and we further investigate how the increase of the number of M^{2}ECs affects the overall performance. In particular, we consider the case in which each community is proxied by 5% to 20% of the community members. Our experimental results show that our selection strategy provides exploitable results. In particular, we observe that our strategy fits for those algorithms detecting a high number of small communities and, at the same time, with those algorithms detecting few big communities. We also that the M^{2}ECs always provide a significant contribution in terms of reduction of average latency and in the data collection capability.

II. Related Work

Multi-access Edge Computing (MEC) is a decentralized cloud technology and point of convergence between telecommunication and Information Technology (IT) services [10]. The MEC paradigm is generally considered a cornerstone of the recent 5G networks [11]. The concept of MEC architecture evolves the standard cloud computing model by flanking to the antennas coupled with radio access network (i.e., base stations) a bunch of powerful, highly virtualized servers that bring the computation closer to peripheral network nodes. In this way. MEC eases the work of such peripheral units by reducing latency for all those latency-dependent applications that run at the edge of the network. To date, MEC implementations consider both indoor and outdoor scenarios [12, 13], covering application fields like traffic [14] and mobility [15]. MCS, instead, is a massive sensing concept emerged in the last few decades from the wide diffusion of mobile and wearables [16].. In recent years, studies on MCS focused on aspects like task assignment [17], energy efficiency [18], and user recruitment techniques [19].

In recent years, researchers have been oriented towards the development of synergies to achieve a close integration between MCS and MEC. Such convergence aims at extending the coverage of traditional MEC solutions, through the activation of mobile MCS devices as support edges for other mobile MCS devices. Fundamental, to plan efficient edge selection strategies in HEC is the concept of community [8]. Community detection techniques may vary on the basis of the definition of community adopted and return very different communities from the same set of elements. During the years, the scientific community has developed several methods to identify communities in dynamic graphs, such as those representing the human mobility over time. In the following, we apply our edge selection strategy to the output of three distinct community detection algorithms suitably selected for their heterogeneity of the identification of communities from user's mobility traces.

III. Background on Community Detection Algorithms

Communities can be detected with algorithms designed to capture the evolution of interactions among users. More specifically, we consider the ParticipAct [3] dataset, which reproduces the human mobility in an urban area in the period

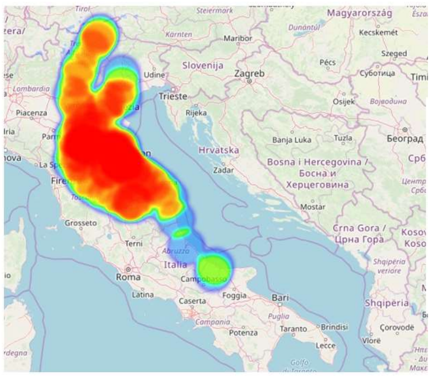

Fig. 1 Graphical representation of ParticipAct.

December 2013 to February 2015. Users are mainly students from the University of Bologna, who were equipped with a smartphone with a pre-installed MCS app. The application reports the user's position at regular intervals, by exploiting the Google Locations APIs. The user's position can be obtained by using information from the GPS, Wi-Fi, or the cellular base stations. We report in Fig. 1 a graphical representation of the ParticipAct dataset. We do not analyze the raw mobility traces, rather we consider the co-location traces. Such traces only report the timestamp at which two users were in proximity. Co-

Algorithm 1 – The M^{2}EC selection strategy

```
Input:   A community set, $\alpha$, $\beta$
Output:  A number of M$^2$ECs
1   Let S be the community set $\{S_1, \dots, S_n\}$ of a given
      Community Detection Algorithm
2   Sort{S} // wlog we assume that $|S_i| \geq |S_j|$ iff $i \geq j$
3   Let SC be the set of the first k communities $\{S_1, \dots, S_k\}$, with
    $k \leq n$, such that:
       - $|S_1 \cup \dots \cup S_k| \geq \alpha$
       - $|S_1 \cup \dots \cup S_{k+1}| - |S_1 \cup \dots \cup S_k| \leq \beta$
4   for all $SC_i \in SC$ do
5     for all $u_i \in SC_i$ do
6        $C_i$ = compute_centrality ($u_i$)
7     end
8     Sort{ $SC_i$} \\ so that $c_i \leq c_j$, $\forall$ c ϵ $C_i$
9   end
10  M$^2$EC_LIST = []  // The output M$^2$EC list initialized as empty
11  // wlog the number of M$^2$ECs is computed as follows
12  N_M$^2$EC = | U | * x  // where U is set of all active users
    $\{u_1, \dots, u_n\}$, and x is the percentage of M$^2$ECs to be
    selected
13  while |M$^2$EC_LIST| < | N_M$^2$EC | do
14    for all $SC_i \in SC$ do
15       // For each community of the community subset SC,
            we select the next M$^2$EC depending on its sorting
16       M$^2$EC_LIST = $SC_i$[next]
17    end
18  end
```

location traces are used to detect communities and, more interestingly, their evolution over time.

We experienced with Infomap [4], TILES [5] and iLCD [7], three algorithms that well capture the evolution of communities along the time. Infomap finds communities of nodes such that the paths connecting them is the shortest possible. Paths are explored by a random walker and, at every step of the walker, the code describing the path is recorded. The random walker stops its process when it is not possible to further minimize the length of such coded-paths. TILES is an online community algorithm that explores the flow of interactions between nodes over time through a domino effect strategy. Such strategy is based on the so-called label propagation procedure. TILES tracks the changes in the neighbor of those nodes that produce a variation in the interaction flow. We also tested iLCD (intrinsic Longitudinal Community Detection), a meta-algorithm that considers the dynamics of a network to detect strongly overlapping communities in a temporal region. Given a co-location trace for a given period (e.g. 1, 6, 12 months), the three algorithms can be configured so that to detect at regular intervals the communities. We refer to such period as Δ. For the purpose of this work, we set Δ = 2 days, so that to obtain a set of communities from every algorithm once every 2 days. We report in Fig. 2, the box plot with the number of communities detected by the three algorithms.

IV. Mobile MEC Selection Strategy

The M^{2}EC selection strategy we propose relies on two main steps. Firstly, we detect communities of users with the community detection algorithms described in Section III. Secondly, we select the M^{2}ECs as representative for each community. Communities are identified by considering the user's mobility of a time period (e.g. 1 month). During such period, we can detect the existing communities with one of 3 algorithms mentioned, at periodic intervals (e.g. daily, weekly, monthly). The number of M^{2}ECs is computed as a percentage of all active users of the platform so to obtain *x*% representatives. The M^{2}EC selection algorithm analyzes the list of communities identified with one of the 3 community detection algorithms. The input list provides for each community an identifier and the list of the community members. The algorithm operates on mobile time windows, without assuming the static behavior of the contacts. It works as follows:

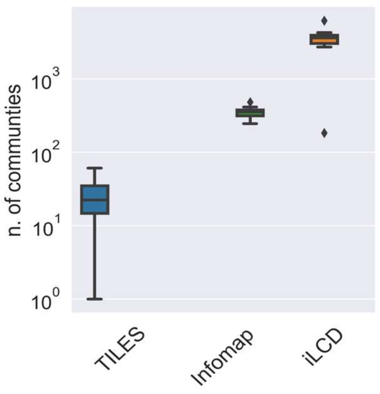

Fig. 2 Number of communities with the 3 algorithms.

TABLE I Experimental Settings

Property	***Value***
City	Bologna
Observation period	March 1st – 31st, 2014
Max n. of participants	133
Number of requests	5k
Δ	1,3, and 7 days
α, β	70%, 2%
M^{2}EC selection strategies	Betweenness, Eigenvector
Percentage of M^{2}ECs	[5-20]%, step 5% step

- To rank the communities on the basis of their cardinality, from the largest to the smallest.
- To select the first k communities according to the following condition: the cardinality of the union of the communities is higher of a given threshold α and the contribution of the k+1 community is lower than a given threshold β. Therefore, the parameter α and β enable to select the optimal number of communities.
- To compute the centrality measure of each community member and to rank the community members according to such measure. For the purpose of this work, we consider the betweenness and the eigenvector centrality measures.
- To compute the number of M^{2}ECs as a percentage of all candidate devices. If the number of M^{2}ECs selected is less or equal to the number of communities selected, than we obtain the list of M^{2}ECs by taking just one representative per community. Otherwise, the algorithm selects the remaining number of M^{2}ECs by selecting one M^{2}EC per community starting from the community with the highest cardinality.

All steps are summarized in Algorithm 1.

V. Experimental Settings and Results

The goal of the experimental settings is to study how the latency and the number of requests satisfied vary as the number of M^{2}ECs increases. Moreover, we also analyze how many M^{2}ECs are required in order to guarantee a given latency requirements and a certain number of requests satisfied. With the term request, we refer to any information that a mobile device requires to upload to the Cloud. We developed a python-based simulation environment that assigns to nodes in the MCS measurement campaign an arbitrary number of requests to be satisfied. A request has an expiration time (Δ) and it can be satisfied in two different ways. Firstly, a request can be satisfied if a node interacts directly with a M^{2}EC through a short-range network interface. Secondly, a request can be satisfied if the node uploads data to the back-end with a broadband connection. During each of the experiments, we simulated a traffic load of

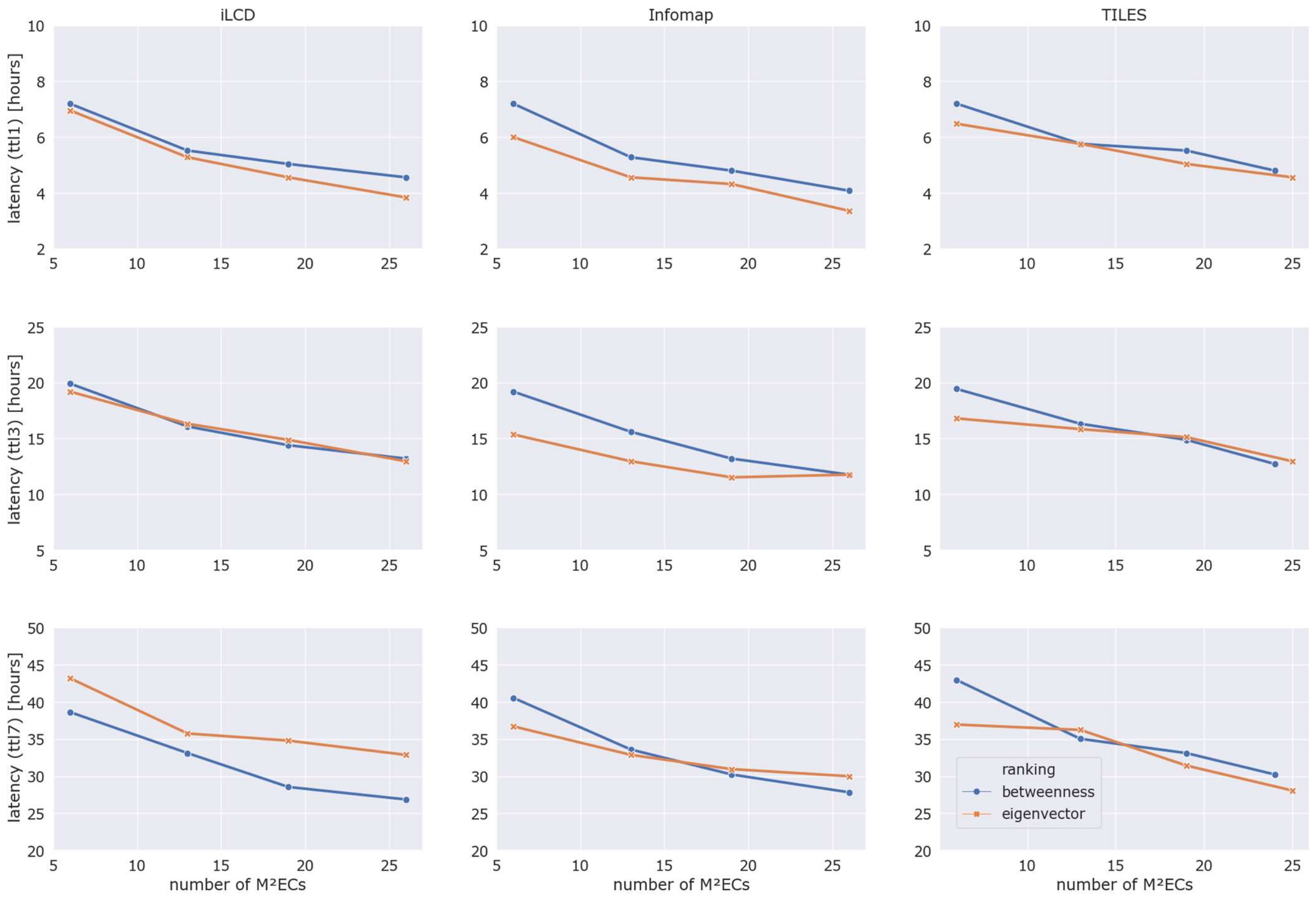

Fig. 3 Average latency by varying the number of M^{2}ECs selected (betweenness and eigenvector)

5k requests, randomly assigned to the MCS nodes. Details of the experimental settings are reported in Table I.

We evaluated on trial some values for the parameters α and β in accordance with the number of communities returned by each of all community detection algorithms considered, and we calibrated the simulator accordingly. Under this respect, Infomap returns at the same time few big communities and many small communities as obtained with iLCD. Differently, TILES returns mainly small communities whose nodes are strongly connected. Such heterogeneity of the communities detected is also reported in Fig. 2, we set α to 70% and β to 2%.

We adopted two centrality measures to rank the community members, namely the betweenness and the eigenvector. On the one hand, the betweenness is a centrality measure based on shortest paths, that assigns to each vertex of the network a value on the basis of the number of shortest paths that pass through it. Such measure reflects the interaction degree that each node has with other nodes of the same network. On the other hand, the eigenvector is a centrality measure that returns the degree of influence that a node exerts within the network, and the score assigned to each node is computed on the basis of the influence of its neighborhood. In other words, the eigenvector assigns higher scores to those nodes that have neighbors with higher connection. We performed several tests by varying the percentage of M^{2}ECs selected and the Δ, ranging from a minimum of 1 to a maximum of 7 days. In all tests performed the percentage of M^{2}ECs selected varies from a minimum of 5% to a maximum of 20% of the total number of active mobiles in the platform. We increase such percentage of 5% step at each test. The graphs in Fig. 3 and Fig. 4 respectively show the results obtained for latency and number of requests satisfied, by varying Δ and the centrality measure. Concerning results in Fig. 3, we observe that the trends for Infomap, iLCD, and TILES follow a regular decreasing trend as the percentage of M^{2}ECs increases. The algorithm selects the best M^{2}ECs with communities returned by iLCD and when the ranking of the community nodes is executed with the eigenvector. Such result is more evident in the line plot for iLCD with Δ 7. The corresponding graph shows an initial better performance when the nodes are ranked with the betweenness (38.6h) with respect to the eigenvector (43.2h). However, once the number of M^{2}ECs increases, the betweenness node ranking allow to reach a latency of 26.9h, whereas the eigenvector node ranking reduces the latency up to 32.9h. We also observe that when the ranking of community nodes is performed with the betweenness, for Δ = 7 the latency is bound between 40h with the 5% of M^{2}ECs and 27.9h with 20% of M^{2}ECs for the Infomap implementation, and between 43h with the minimum number of M^{2}ECs selected to 30.2h with 20% of M^{2}ECs for the

TILES implementation. We notice that that when the algorithm runs upon the communities discovered by the TILES, it is not able to select as many M^{2}ECs as the ones selected within the communities discovered by Infomap and iLCD. Such behavior is mainly due to the restricted number of communities discovered by TILES and the percentage of nodes belonging to each community with respect to the total number of nodes of the population (less than 48% of the total). The results obtained by setting the Δ to 3 days show that the eigenvector returns the best latency results with all the three community detection algorithms, that present very similar decreasing latency trends. Concerning the number of requests satisfied (Fig. 4), we observe that all the trends of the line-plot matrix show an increasing trend independently of the node ranking centrality measure in use for all community detection algorithms. For $\Delta = 7$ we register that our algorithm selects the best M^{2}ECs with the communities discovered by TILES and when the node ranking is performed with the betweenness. In fact, as the number of M^{2}ECs increases we register for the TILES implementation a number of requests satisfied that ranges from 3415 (68%) with 5% of M^{2}ECs up to 4081 (82%) with 20% of M^{2}ECs. Always for $\Delta = 7$ we observe that the iLCD implementation performs good with a low percentage of M^{2}ECs selected, returning 3543 (71%) requests satisfied with 5% of M^{2}ECs. We also observe that for iLCD the performance improve slowly as the number of M^{2}ECs increases. In fact, with 20% of M^{2}ECs selected the iLCD implementation satisfies 4070 (81%) requests, returning an improvement of only 10% with respect to the performance of 5% M^{2}ECs. Finally, we observe that as the number of M^{2}ECs selected increases the node ranking with the eigenvector performs as good as the betweenness one only with TILES in any of the Δs considered (59%, 71%, and 82% respectively for $\Delta = 1$, $\Delta = 3$, and $\Delta = 7$).

VI. Conclusions

Mobile edge computing technologies are nowadays widely used in many applications. In the case of CrowdSensing they are particularly effective as edges may act as localized data collectors and aggregators. A further, potential innovation in the integration of MEC and MCS is the extension of the concept of edges by introducing mobile edges (M^{2}ECs) implemented by the devices that are part of the CrowdSensing platform itself. This idea gave recently rise to a research trend, in which the key questions concern the real advantages of using mobile edges, the applicative scenarios in which are more valuable, and the overheads or penalties given by their use. One aspect that is already clear is that, since mobile edges leverage on opportunistic communications to collect data from the other devices, the applicative scenario can be one in which communication latencies are not critical (this is typical for

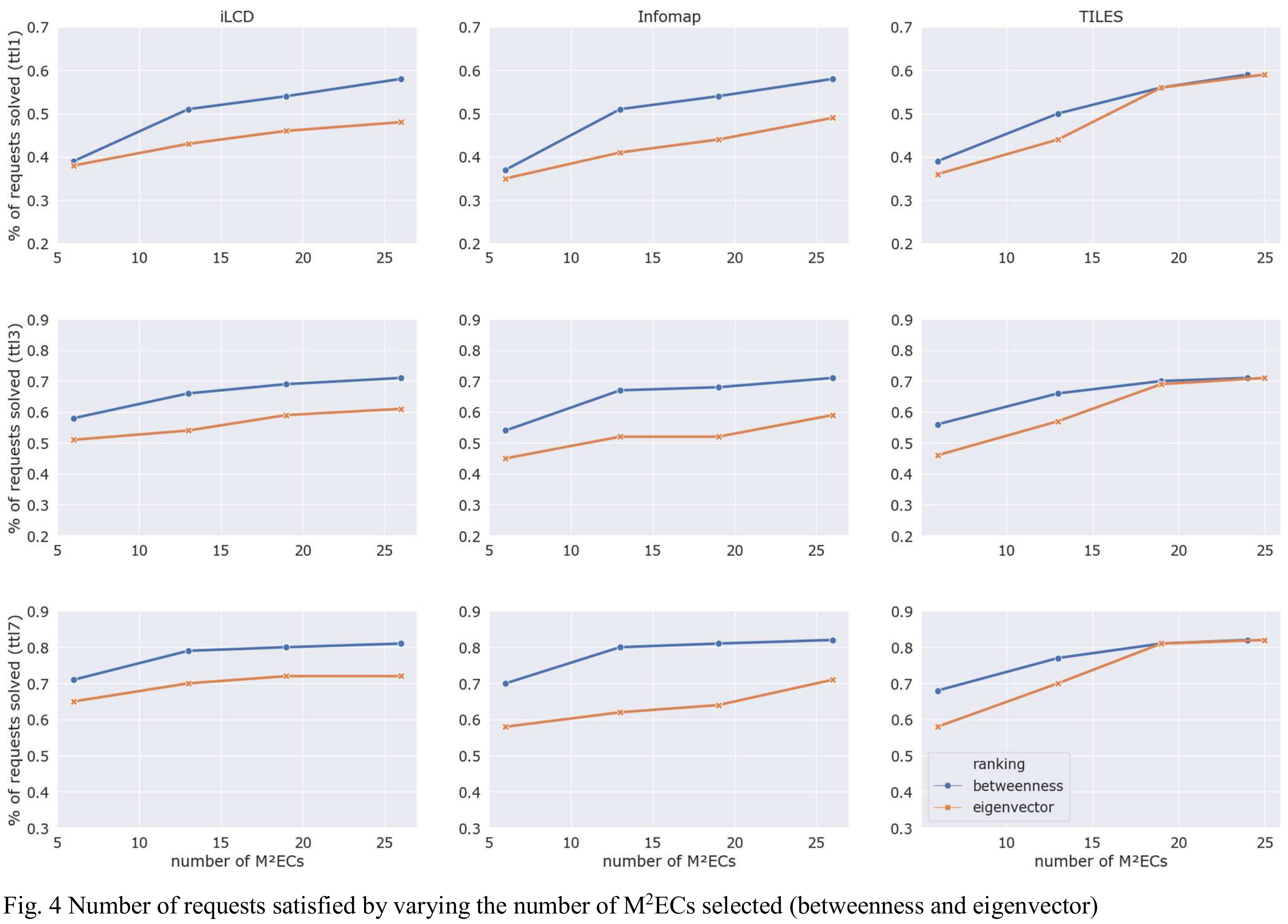

Fig. 4 Number of requests satisfied by varying the number of M^{2}ECs selected (betweenness and eigenvector)

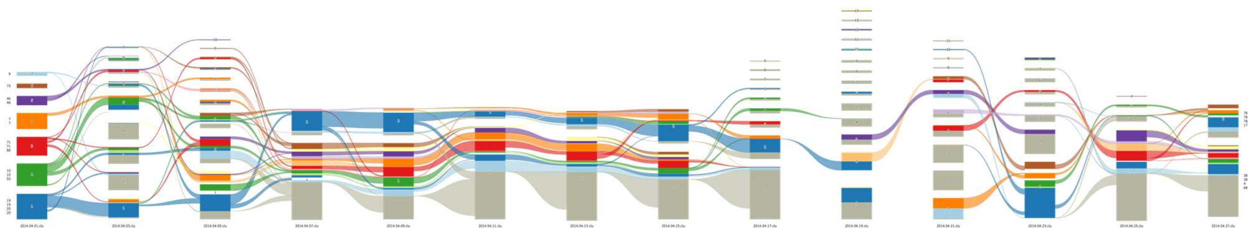

Fig. 5 Example of evolution of communities for a time period.

applications intended for offline analysis of the collected data). However, even if latency (defined as the time elapsed from the time in which the data is produced to the time in which the data is available at the data center) is not critical, it still plays a role since its dimension determines the boundary of application of mobile edges in MCS. This work addresses this specific aspect as it aims at estimating the impact on the latency due to two different architectural approaches in the design of the mobile edges solution (which are the number of M^2ECs to use and the Δ of the data before they are sent to the data center through broadband connection). The results have shown that, although the use of a larger number of M^2ECs provides a reduction of the latency, the strongest reduction is obtained by reducing the Δ. However, even when the Δ is significantly reduced, the M^2ECs still play an important role in the data collection as they are able to collect a fraction of data ranging from 50% to 80% in most of the cases. This result, that was not expected, is related to the "social" nature of M^2ECs that are chosen based on the evolving communities of devices as reported with the alluvial graph in Fig. 5. In practice, this result implicitly suggests that even when the number of M^2ECs is low they are able to collect a large fraction of data from their communities, while the devices that have few connections with the others and that are out of the communities will have to transmit the data by leveraging on their broadband connections.

Another indirect result of this work is that the algorithm used for the detection of the communities in the selection of the M^2ECs has a minor but observable effect on the latency. In fact, the different community detection algorithms exhibit a slight difference in performance, especially when the number of M^2ECs is low. Furthermore, the similar trends showed in Fig. 3 and Fig. 4 suggest that our M^2EC selection algorithm performs well with community detection algorithms that return community sets made up of very few numerous communities, as well as with community detection algorithms that return community sets made up of many communities with very strongly connected nodes. In the future we are planning to extend this analysis by considering different ways for computing the communities and by analyzing other parameters, like the number of communities for which a M^2EC is selected.

REFERENCES

[1] D. Belli, S. Chessa, A. Corradi, G. Di Paolo, L. Foschini and M. Girolami, "Selection of Mobile Edges for a Hybrid CrowdSensing Architecture," 2019 IEEE Symposium on Computers and Communications (ISCC), Barcelona, Spain, 2019, pp. 1-6.

[2] P. Bellavista, D. Belli, S. Chessa, L. Foschini, "A social-driven edge computing architecture for mobile crowdsensing management", *IEEE Communications Magazine*, 57(4), pp. 68-73, 2019.

[3] G. Cardone, A. Corradi, L. Foschini, R. Ianniello, "ParticipAct: A large-scale crowdsensing platform", *IEEE Transactions on Emerging Topics in Computing*, 4(1), pp. 21-32, May, 2015.

[4] D. Edler, L. Bohlin, M. Rosvall, "Mapping higher-order network flows in memory and multilayer networks with infomap", *Algorithms*, 10(4), 112, 2017.

[5] G. Rossetti, L. Pappalardo, D. Pedreschi, F. Giannotti, "TILES: an online algorithm for community discovery in dynamic social networks", *Machine Learning*, 106(8), pp. 1213-1241, 2017.

[6] D. Belli, S. Chessa, A. Corradi, L. Foschini, M. Girolami, Optimization strategies for the selection of mobile edges in hybrid crowdsensing architectures, Computer Communications, Volume 157, 2020, Pages 132-142,

[7] R. Cazabet, F. Amblard, C. Hanachi, "Detection of overlapping communities in dynamic social networks", in IEEE Second International Conference on Social Computing, pp. 309-314, 2010.

[8] G. Rossetti, R. Cazabet, "Community discovery in dynamic networks: a survey", ACM Computing Surveys (CSUR), 51(2), 35, 2018.

[9] M. Girolami, P. Barsocchi, S. Chessa and F. Furfari, "A social-based service discovery protocol for mobile Ad Hoc networks," *2013 12th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET)*, Ajaccio, 2013, pp. 103-110.

[10] A. Ndikumana, N. H. Tran, T. M. Ho, Z. Han, W. Saad, D. Niyato, C. S. Hong, "Joint communication, computation, caching, and control in big data multi-access edge computing", *IEEE Transactions on mobile Computing*, 2019.

[11] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, V. Young, "mobile edge computing – a key technology towards 5G", *ETSI White Paper*, 11(11), pp. 1-16, 2015.

[12] J. L. Carrera Villacrés, Z. Zaho, M. Wenger, T. Braun, "MEC-based UWB Indoor Tracking System", in *15th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*, 2019.

[13] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: Concepts, methodologies, and applications", ACM Transactions on Intelligent Systems and Technology, vol. 5, no. 3, pp. 1–55, Sep 2014.

[14] H. Peng, Q. Ye, X. Shen, "Spectrum management for multi-access edge computing in autonomous vehicular networks", *IEEE Transactions on Intelligent Transportation Systems*, pp. 1-12, 2019.

[15] P. Zhang, M. Durresi, A. Durresi, "Multi-access edge computing aided mobility for privacy protection in internet of things", *Computing*, 101(7), pp. 729-742, 2019.

[16] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges", *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32-39, 2011.

[17] W. Gong, B. Zhang, C. Li, "Task assignment in mobile crowdsensing: Present and future directions", *IEEE Network*, 32(4), pp. 100-107, 2018.

[18] H. Xiong, D. Zhang, L. Wang, J. P. Gibson, J. Zhu, "EEMC: Enabling energy-efficient mobile crowdensing with anonymous participants", *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 6, no. 3, pp. 1-26, 2015.

D. Belli, S. Chessa, B. Kantarci, L. Foschini, "A capacity-aware user recruitment framework for fog-based mobile crowdsensing", *IEEE Symposium on Computers and Communications (ISCC)*, 2019.